

This is a summary of the original paper, entitled “Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case” which appears in HPCA 2015 [64].

Adaptive-Latency DRAM (AL-DRAM)

Donghyuk Lee Yoongu Kim Gennady Pekhimenko
Samira Khan Vivek Seshadri Kevin Chang Onur Mutlu
Carnegie Mellon University

Abstract

This paper summarizes the idea of Adaptive-Latency DRAM (AL-DRAM), which was published in HPCA 2015 [64]. The key goal of AL-DRAM is to exploit the extra margin that is built into the DRAM timing parameters to reduce DRAM latency. The key observation is that the timing parameters are dictated by the worst-case temperatures and worst-case DRAM cells, both of which lead to small amount of charge storage and hence high access latency. One can therefore reduce latency by adapting the timing parameters to the current operating temperature and the current DIMM that is being accessed. Using an FPGA-based testing platform, our work first characterizes the extra margin for 115 DRAM modules from three major manufacturers. The experimental results demonstrate that it is possible to reduce four of the most critical timing parameters by a minimum/maximum of 17.3%/54.8% at 55°C while maintaining reliable operation. AL-DRAM adaptively selects between multiple different timing parameters for each DRAM module based on its current operating condition. AL-DRAM does not require any changes to the DRAM chip or its interface; it only requires multiple different timing parameters to be specified and supported by the memory controller. Real system evaluations show that AL-DRAM improves the performance of memory-intensive workloads by an average of 14% without introducing any errors [64].

1. Summary

1.1. Problem: High DRAM Latency

A DRAM chip is made of capacitor-based cells that represent data in the form of electrical charge. To store data in a cell, charge is injected, whereas to retrieve data from a cell, charge is extracted. Such *movement of charge* happens through a wire called *bitline*. Due to the large resistance and the large capacitance of the bitline, it takes long time to access DRAM cells. To guarantee correct operation, DRAM manufacturers impose a set of minimum latency restrictions on DRAM accesses, called *timing parameters* [44]. Ideally, timing parameters should provide *just enough* time for a DRAM chip to operate correctly. In practice, however, there is a very large margin in the timing parameters to ensure correct operation under *worst-case* conditions with respect to two aspects. First, due to *process variation*, some outlier cells suffer from a larger RC-delay than other cells [46, 67], and require more time to be accessed. Second, due to *temperature dependence*, DRAM cells lose more charge at high temperature [124], and therefore require more time to be sensed and restored. Due to the worst-case pro-

visioning of timing parameters, it takes longer time to access most of DRAM under most operating conditions than necessary for correct operation.

1.2. Key Observations and Our Goal

Most DRAM chips do *not* contain the worst-case cell with the largest latency. Using an FPGA-based testing platform, we profile 115 DRAM modules and observe that the slowest cell, having the smallest amount of charge, for a typical chip is still faster than that of the worst-case chip. We expose the large margin built into DRAM timing parameters. In particular, we identify four timing parameters that are the most critical during a DRAM access: t_{RCD} , t_{RAS} , t_{WR} , and t_{RP} . At 55°C, we demonstrate that the parameters can be reduced by an average of 17.3%, 37.7%, 54.8%, and 35.2% while still maintaining correctness.

Most DRAM chips are *not* exposed to the worst-case temperature of 85°C. We measure the DRAM ambient temperature in a server cluster running a very memory-intensive benchmark, and find that the temperature *never* exceeds 34°C — as well as never changing by more than 0.1°C per second. Other works [38, 39, 71] also observed that worst-case DRAM temperatures are not common and servers operate at much lower temperatures [38, 39, 71].

Based on these observations, we show *that* typical DRAM chips operating at typical temperatures (e.g., 55°C) are capable of providing a much smaller access latency, but are nevertheless forced to operate at the largest latency of the worst-case due to the use of only a single set of timing parameters dictated by the worst case.

Our goal in our HPCA 2015 paper [64] is to exploit the extra margin that is built into the DRAM timing parameters to reduce DRAM latency and thus improve performance. To this end, we first provide a detailed analysis of why we can reduce DRAM timing parameters without sacrificing reliability.

1.3. Charge & Latency Interdependence

The operation of a DRAM cell is governed by two important parameters: *i)* the quantity of charge and *ii)* the latency it takes to move charge. These two parameters are closely related to each other. Based on SPICE simulations with a detailed DRAM model, we identify the quantitative relationship between charge and latency [64]. While our HPCA 2015 paper provides the detailed analyses of this relationship, here we summarize the three key observations. First, having more charge in a DRAM cell accelerates the sensing operation in the cell, especially at the beginning of sensing, enabling the opportunity to shorten the corresponding timing parameters (t_{RCD} and t_{RAS}). Second, when restoring the charge in a DRAM

cell, a large amount of the time is spent on injecting the final small amount of charge into the cell. If there is already enough charge in the cell for the next access, the cell does not need to be fully restored. In this case, it is possible to shorten the latter part of the restoration time, creating the opportunity to shorten the corresponding timing parameters (τ_{RAS} and τ_{WR}). Third, at the end of precharging, i.e., setting the bitline into the initial voltage level (before accessing a cell) for the next access, a large amount of the time is spent on precharging the final small amount of bitline voltage difference from the initial level. When there is already enough charge in the cell to overcome the voltage difference in the bitline, the bitline does not need to be fully precharged. Thus, it is possible to shorten the final part of the precharge time, creating the opportunity to shorten the corresponding timing parameter (τ_{RP}). Based on these three observations, we understand that *timing parameters can be shortened if DRAM cells have enough charge*.

1.4. Adaptive-Latency DRAM

As explained, the amount of charge in the cell right before an access to it plays a critical role in whether the correct data is retrieved from the cell. In Figure 1, we illustrate the impact of process variation using two different cells: one is a *typical* cell (left column) and the other is the worst-case cell which deviates the most from the typical (right column). The worst-case cell contains less charge than the typical cell in its initial state. This is because of two reasons. First, due to its *large resistance*, the worst-case cell cannot allow charge to flow inside quickly. Second, due to its *small capacitance*, the worst-case cell cannot store much charge even when it is full. To accommodate such a worst-case cell, existing timing parameters are set to a large value.

In Figure 1, we also illustrate the impact of temperature dependence using two cells at two different temperatures: *i*) a typical temperature (55°C, bottom row), and *ii*) the worst-case temperature (85°C, top row) supported by DRAM standards. Both typical and worst-case cells leak charge at a faster rate at the worst-case temperature. Therefore, not only does the worst-case cell have less charge to begin with, but it is left with *even less* charge at the worst temperature because it leaks charge at a faster rate (top-right in Figure 1). To accommodate the combined effect of process variation *and* temperature dependence, existing timing parameters are set to a very large value. That is why the worst-case condition for correctness is specified by the top-right of Figure 1, which shows the least amount of charge stored in the *worst-case cell at the worst-case temperature* in its initial state. On top of this, DRAM manufacturers still add an extra latency margin even for such worst-case conditions. In other words, the amount of charge at the worst-case condition is still greater than what is required for correctness under that condition.

If we were to reduce the timing parameters, we would also be reducing the charge stored in the cells. It is important to note, however, that we are proposing to exploit *only the additional slack* (in terms of charge) compared to the worst-case. This allows us to provide as strong of a reliability guarantee as the worst-case. In Figure 1, we illustrate the impact of reducing the timing parameters. The lightened portions inside the cells represent the amount of charge that we are giving up by using the reduced timing parameters. Note that we are not giving up

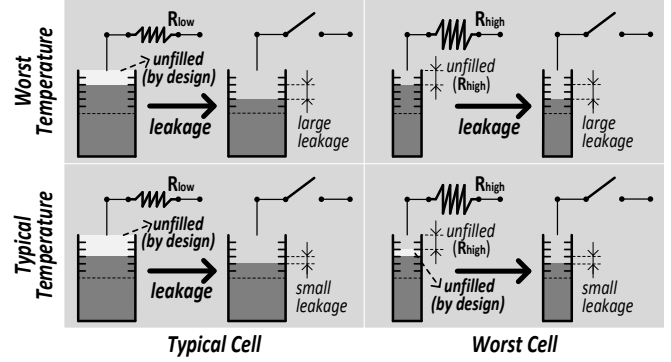


Figure 1: Effect of Reduced Latency: Typical vs. Worst

any charge for the worst-case cell at the worst-case temperature. Although the other three cells are not fully charged in their initial state, they are left with a similar amount of charge as the worst-case (top-right). This is because these cells are capable of either holding more charge (typical cell, left column) or holding their charge longer (typical temperature, bottom row). Therefore, optimizing the timing parameters (based on the amount of existing charge slack) provides the opportunity to reduce overall DRAM latency while still maintaining the reliability guarantees provided by the DRAM manufacturers.

Based on these observations, we propose Adaptive-Latency DRAM (AL-DRAM), a mechanism that dynamically optimizes the timing parameters for different modules at different temperatures. AL-DRAM exploits the *additional charge slack* present in the common-case compared to the worst-case, thereby preserving the level of reliability (at least as high as the worst-case) provided by DRAM manufacturers.

1.5. DRAM Latency Profiling

We present and analyze the results of our DRAM profiling experiments, performed on our FPGA-based DRAM testing infrastructure [22, 48, 49, 50, 64, 69, 99]. Figures 2a and 2b show the results of this experiment for the read and write latency tests. The y-axis plots the sum of the relevant timing parameters (τ_{RCD} , τ_{RAS} , and τ_{RP} for the read latency test and τ_{RCD} , τ_{WR} , and τ_{RP} for the write latency test). The solid black line shows the latency sum of the standard timing parameters (DDR3 DRAM specification). The dotted red line and the dotted blue line show the acceptable latency parameters that do not cause any errors for each DIMM at 85°C and 55°C, respectively. The solid red line and blue line show the average acceptable latency across all DIMMs.

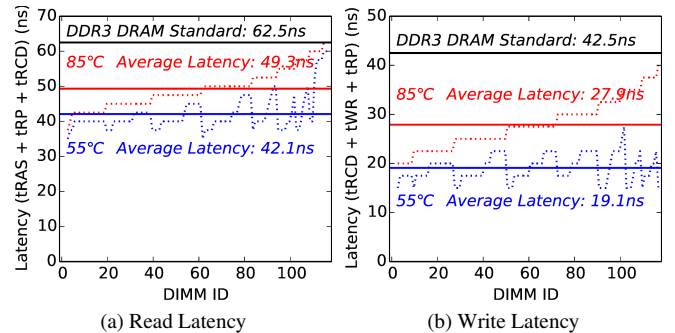


Figure 2: Access Latency Analysis of 115 DIMMs

We make two observations. First, even at the highest temperature of 85°C, DIMMs have a high potential of access with reduced latency: 21.1% on average for read, and 34.4% on average for write operations. This is a direct result of the possible reductions in timing parameters $t_{RCD}/t_{RAS}/t_{WR}/t_{RP}$ — 15.6%/20.4%/20.6%/28.5% on average across all the DIMMs. As a result, we conclude that process variation and lower temperatures enable a significant potential to reduce DRAM access latencies. Second, we observe that at lower temperatures (e.g., 55°C), the potential for latency reduction is even greater (32.7% on average for read, and 55.1% on average for write operations), where the corresponding reduction in timing parameters $t_{RCD}/t_{RAS}/t_{WR}/t_{RP}$ are 17.3%/37.7%/54.8%/35.2% on average across all the DIMMs.

1.6. Real-System Evaluation

We evaluate AL-DRAM on a real system that offers dynamic software-based control over DRAM timing parameters at run-time [9, 10]. We use the minimum values of the timing parameters that do not introduce any errors at 55°C for any module to determine the latency reduction at 55°C. Thus, the latency is reduced by 27%/32%/33%/18% for $t_{RCD}/t_{RAS}/t_{WR}/t_{RP}$, respectively. Our full methodology is described in our HPCA 2015 paper [64].

Figure 3 shows the performance improvement of reducing the timing parameters in the evaluated memory system with one rank and one memory channel at 55°C operating temperature. We run a variety of different applications in two different configurations. The first one (single-core) runs only one thread, and the second one (multi-core) runs multiple applications/threads. We run each configuration 30 times (only SPEC benchmarks are executed 3 times due to their large execution times), and present the average performance improvement across all the runs and their standard deviation as an error bar. Based on the last-level cache misses per kilo instructions (MPKI), we categorize our applications into memory-intensive or non-intensive groups, and report the geometric mean performance improvement across all applications from each group.

We draw three key conclusions from Figure 3. First, AL-DRAM provides significant performance improvement over the baseline (as high as 20.5% for the very memory-bandwidth-intensive STREAM applications [78]). Second, when the memory system is under higher pressure with multi-core/multi-threaded applications, we observe significantly higher performance (than in the single-core case) across all applications from our workload pool. Third, as expected, memory-intensive applications benefit more in performance than non-memory-intensive workloads (14.0% vs. 2.9% on average). We conclude that by reducing the DRAM timing parameters using AL-DRAM, we can speed up a real system by 10.5% (on average across all 35 workloads on the multi-core/multi-thread configuration).

1.7. Other Results and Analyses in Our Paper

Our HPCA paper includes more DRAM latency analyses and system performance evaluations.

- **Effect of Changing the Refresh Interval on DRAM Latency.** We evaluate DRAM latency at different refresh intervals. We observe that refreshing DRAM cells more frequently enables more DRAM latency reduction.

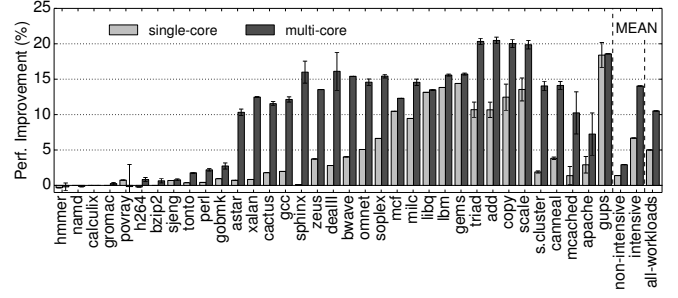


Figure 3: Real System Performance Improvement with AL-DRAM

- **Effect of Reducing Multiple Timing Parameters.** We study the potential for reducing multiple timing parameters simultaneously. Our key observation is that reducing one timing parameter leads to decreasing the opportunity to reduce another timing parameter simultaneously.
- **Analysis of the Repeatability of Cell Failures.** We perform tests for five different scenarios to determine that a cell failure due to reduced latency is repeatable: same test, test with different data patterns, test with timing-parameter combinations, test with different temperatures, and read/write test. Most of these scenarios show that a very high fraction (more than 95%) of the erroneous cells consistently experience an error over multiple iterations of the same test.
- **Performance Sensitivity Analyses.** We analyze the impact of increasing the number of ranks and channels, executing heterogeneous workloads, using different row buffer policies.

2. Significance

2.1. Novelty

To our knowledge, our HPCA 2015 paper is the first work to *i)* provide a detailed qualitative and empirical analysis of the relationship between *process variation* and *temperature dependence* of modern DRAM devices on the one side, and DRAM access latency on the other side (we directly attribute the relationship between the two to *the amount of charge* in cells), *ii)* experimentally characterize a large number of existing DIMMs to understand the potential of reducing DRAM timing constraints, *iii)* provide a practical mechanism that can take advantage of this potential, and *iv)* evaluate the performance benefits of this mechanism by *dynamically optimizing* DRAM timing parameters on a real system using a variety of real workloads. We make the following major contributions.

Addressing a Critical Real Problem, High DRAM Latency, with Low Cost. High DRAM latency is a critical bottleneck for overall system performance in a variety of modern computing systems [81, 90], especially in real large-scale server systems [72]. Considering the difficulties in DRAM scaling [46, 81, 90], the problem is getting worse in future systems due to process variation. Our HPCA 2015 work leverages the heterogeneity created by DRAM process variation across DRAM chips and system operating conditions to mitigate the DRAM latency problem. We propose a practical mechanism, *Adaptive-Latency DRAM*, which mitigates DRAM latency with very modest hardware cost, and with *no changes* to the DRAM chip itself.

Low Latency DRAM Architectures. Previous works [23, 24, 42, 53, 65, 77, 82, 105, 108, 112, 129] propose new DRAM architectures that provide lower latency. These works improve DRAM latency at the cost of either significant additional DRAM chip area (i.e., extra sense amplifiers [77, 105, 112], an additional SRAM cache [42, 129]), specialized protocols [23, 53, 65, 108] or a combination of these. Our proposed mechanism requires *no changes* to the DRAM chip and the DRAM interface, and hence has almost negligible overhead. Furthermore, AL-DRAM is largely orthogonal to these proposed designs, and can be applied in conjunction with them, providing greater cumulative reduction in latency.

Large-Scale Latency Profiling of Modern DRAM Chips. Using our FPGA-based DRAM testing infrastructure [22, 48, 49, 50, 64, 69, 99], we profile 115 DRAM modules (862 DRAM chips in total) and show that there is significant timing variation between different DIMMs at different temperatures. We believe that our results are statistically significant to validate our hypothesis that the DRAM timing parameters strongly depend on the amount of cell charge. We provide detailed characterization of each DIMM online at the SAFARI Research Group website [63]. Furthermore, we introduce our FPGA-based DRAM infrastructure and experimental methodology for DRAM profiling, which are carefully constructed to represent the worst-case conditions in power noise, bitline/wordline coupling, data patterns, and access patterns. Such information will hopefully be useful for future DRAM research.

Extensive Real System Evaluation of DRAM Latency. We evaluate our mechanism on a real system and show that our mechanism provides significant performance improvement. Reducing the timing parameters strips the excessive margin in DRAM’s electrical charge. We show that the remaining margin is *enough* for DRAM to operate correctly. To verify the correctness of our experiments, we ran our workloads for 33 days non-stop, and examined their and the system’s correctness with reduced timing parameters. Using the reduced timing parameters over the course of 33 days, our real system was able to execute 35 different workloads in both single-core and multi-core configurations while preserving correctness and being *error-free*. Note that these results do *not* absolutely guarantee that no errors can be introduced by reducing the timing parameters. However, we believe that we have demonstrated a proof-of-concept which shows that DRAM latency can be reduced at no impact on DRAM reliability. Ultimately, the DRAM manufacturers can provide the reliable timing parameters for different operating conditions and modules.

Other Methods for Lowering Memory Latency. There are many works that reduce *overall memory access latency* by modifying DRAM, the DRAM-controller interface, and DRAM controllers. These works enable more parallelism and bandwidth [4, 5, 23, 24, 53, 62, 108, 120, 127, 131], reduce refresh counts [48, 69, 70, 99, 119], accelerate bulk operations [24, 107, 108, 109], accelerate computation in the logic layer of 3D-stacked DRAM [2, 3, 40, 126], enable better communication between CPU and other devices through DRAM [66], leverage DRAM access patterns [41], reduce write-related latencies by better designing DRAM and DRAM control policies [25, 58, 106], reduce overall queuing latencies

in DRAM by better scheduling memory requests [11, 12, 28, 35, 43, 45, 51, 52, 59, 60, 61, 78, 79, 80, 86, 87, 92, 104, 114, 115, 116, 117, 118, 130], employing prefetching [6, 21, 26, 27, 32, 34, 36, 37, 59, 83, 84, 85, 88, 89, 91, 93, 113], memory/cache compression [1, 7, 8, 29, 31, 33, 94, 95, 96, 97, 111, 121, 128], or better caching [47, 100, 101, 110]. Our proposal is orthogonal to all of these approaches and can be applied in conjunction with them to achieve even higher latency reductions.

2.2. Potential Long-Term Impact

Tolerating High DRAM Latency by Exploiting DRAM Intrinsic Characteristics. Today, there is a large latency cliff between the on-chip last level cache and off-chip DRAM, leading to a large performance fall-off when applications start missing in the last level cache. By enabling lower DRAM latency, our mechanism, Adaptive-Latency DRAM, smoothens this latency cliff without adding another layer into the memory hierarchy.

Applicability to Future Memory Devices. We show the benefits of the common-case timing optimization in modern DRAM devices by taking advantage of intrinsic characteristics of DRAM. Considering that most memory devices adopt a unified specification that is dictated by the worst-case operating condition, our approach that optimizes device latency for the common case can be applicable to other memory devices by leveraging the intrinsic characteristics of the technology they are built with. We believe there is significant potential for approaches that could reduce the latency of Phase Change Memory (PCM) [30, 55, 56, 57, 75, 98, 102, 103, 123, 125], STT-MRAM [54, 68, 75], RRAM [122], and Flash memory [13, 14, 15, 16, 17, 18, 19, 20, 73, 74, 76].

New Research Opportunities. Adaptive-Latency DRAM creates new opportunities by enabling mechanisms that can leverage the heterogeneous latency offered by our mechanism. We describe a few of these briefly.

Optimizing the operating conditions for faster DRAM access. Adaptive-Latency DRAM provides different access latency at different operating conditions. Thus, optimizing the DRAM operating conditions enables faster DRAM access with Adaptive-Latency DRAM. For instance, balancing DRAM accesses over different DRAM channels and ranks leads to reducing the DRAM operating temperature, maximizing the benefits from Adaptive-Latency DRAM. At the system level, operating the system at a constant low temperature can enable the use of AL-DRAM’s lower latency more frequently.

Optimizing data placement for reducing overall DRAM access latency. We characterize the latency variation in different DIMMs due to process variation. Placing data based on this information and the latency criticality of data maximizes the benefits of lowering DRAM latency.

Error-correction mechanisms to further reduce DRAM latency. Error-correction mechanisms can fix the errors from lowering DRAM latency even further, leading to further reduction in DRAM latency without errors. Future research that uses error correction to enable even lower latency DRAM is therefore promising as it opens a new set of trade-offs.

References

- [1] B. Abali et al. Memory Expansion Technology (MXT): Software support and performance. In *IBM Journal of Research and Development*, 2001.
- [2] J. Ahn et al. A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing. In *ISCA*, 2015.
- [3] J. Ahn et al. PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture. In *ISCA*, 2015.
- [4] J. H. Ahn et al. Improving System Energy Efficiency with Memory Rank Subsetting. In *ACM TACO*, 2012.
- [5] J. H. Ahn et al. Multicore DIMM: an Energy Efficient Memory Module with Independently Controlled DRAMs. In *IEEE CAL*, 2009.
- [6] A. Alameldeen and D. Wood. Interactions Between Compression and Prefetching in Chip Multiprocessors. In *HPCA*, 2007.
- [7] A. R. Alameldeen and D. A. Wood. Adaptive Cache Compression for High-Performance Processors. In *ISCA*, 2004.
- [8] A. R. Alameldeen and D. A. Wood. Frequent Pattern Compression: A Significance-Based Compression Scheme for L2 Caches. In *Technical Report, UM-Madison 1599, University of Wisconsin-Madison*, 2004.
- [9] AMD. *AMD Opteron 4300 Series processors*. <http://www.amd.com/en-us/products/server/4000/4300>.
- [10] AMD. BKDG for AMD Family 16h Models 00h-0Fh Processors, 2013.
- [11] R. Ausavarungrun et al. Staged memory scheduling: achieving high performance and scalability in heterogeneous systems. In *ISCA*, 2012.
- [12] R. Ausavarungrun et al. Exploiting Inter-Warp Heterogeneity to Improve GPGPU Performance. In *PACT*, 2015.
- [13] Y. Cai et al. Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis. In *DATE*, 2012.
- [14] Y. Cai et al. Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling. In *DATE*, 2013.
- [15] Y. Cai et al. Read Disturb Errors in MLC NAND Flash Memory: Characterization, Mitigation, and Recovery. In *DSN*, 2015.
- [16] Y. Cai et al. Data retention in MLC NAND flash memory: Characterization, optimization, and recovery. In *HPCA*, 2015.
- [17] Y. Cai et al. Program Interference in MLC NAND Flash Memory: Characterization, Modeling, and Mitigation. In *ICCD*, 2013.
- [18] Y. Cai et al. Flash correct-and-refresh: Retention-aware error management for increased flash memory lifetime. In *ICCD*, 2012.
- [19] Y. Cai et al. Error Analysis and Retention-Aware Error Management for NAND Flash Memory. In *ITJ*, 2013.
- [20] Y. Cai et al. Neighbor-cell Assisted Error Correction for MLC NAND Flash Memories. In *SIGMETRICS*, 2014.
- [21] P. Cao et al. A Study of Integrated Prefetching and Caching Strategies. In *SIGMETRICS*, 1995.
- [22] K. Chang et al. Understanding Latency Variation in Modern DRAM Chips: Experimental Characterization, Analysis, and Optimization. In *SIGMETRICS*, 2016.
- [23] K. Chang et al. Improving DRAM performance by parallelizing refreshes with accesses. In *HPCA*, 2014.
- [24] K. Chang et al. Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM. In *HPCA*, 2016.
- [25] N. Chatterjee et al. Staged Reads: Mitigating the Impact of DRAM Writes on DRAM Reads. In *HPCA*, 2012.
- [26] R. Cooksey et al. A Stateless, Content-directed Data Prefetching Mechanism. In *ASPLOS*, 2002.
- [27] F. Dahlgren et al. Sequential Hardware Prefetching in Shared-Memory Multiprocessors. In *IEEE TPDS*, 1995.
- [28] R. Das et al. Application-to-core mapping policies to reduce memory system interference in multi-core systems. In *HPCA*, 2013.
- [29] R. de Castro et al. Adaptive compressed caching: design and implementation. In *SBAC-PAD*, 2003.
- [30] G. Dhiman et al. PDRAM: A hybrid PRAM and DRAM main memory system. In *DAC*, 2009.
- [31] F. Douglass. The Compression Cache: Using On-line Compression to Extend Physical Memory. In *Winter USENIX Conference*, 1993.
- [32] J. Dundas and T. Mudge. Improving Data Cache Performance by Pre-executing Instructions Under a Cache Miss. In *ICS*, 1997.
- [33] J. Dussier et al. Zero-content Augmented Caches. In *ICS*, 2009.
- [34] E. Ebrahimi et al. Prefetch-aware Shared Resource Management for Multi-core Systems. In *ISCA*, 2011.
- [35] E. Ebrahimi et al. Parallel application memory scheduling. In *MICRO*, 2011.
- [36] E. Ebrahimi et al. Coordinated Control of Multiple Prefetchers in Multi-core Systems. In *MICRO*, 2009.
- [37] E. Ebrahimi et al. Techniques for bandwidth-efficient prefetching of linked data structures in hybrid prefetching systems. In *HPCA*, 2009.
- [38] N. El-Sayed et al. Temperature Management in Data Centers: Why Some (Might) Like It Hot. In *SIGMETRICS*, 2012.
- [39] N. El-Sayed et al. Temperature Management in Data Centers: Why Some (Might) Like It Hot. In *Technical Report, CSRG-615, University of Toronto*, 2012.
- [40] Q. Guo et al. 3D-Stacked Memory-Side Acceleration: Accelerator and System Design. In *WONDP*, 2014.
- [41] H. Hassan et al. ChargeCache: Reducing DRAM Latency by Exploiting Row Access Locality. In *HPCA*, 2016.
- [42] H. Hidaka et al. The Cache DRAM Architecture: A DRAM with an On-Chip Cache Memory. In *IEEE Micro*, 1990.
- [43] E. Ipek et al. Self-optimizing memory controllers: A reinforcement learning approach. In *ISCA*, 2008.
- [44] JEDEC. *Standard No. 79-3F. DDR3 SDRAM Specification*, July 2012.
- [45] A. Jog et al. Exploiting Core-Criticality for Enhanced GPU Performance. In *SIGMETRICS*, 2016.
- [46] U. Kang et al. Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling. In *The Memory Forum*, 2014.
- [47] S. Khan et al. Improving Cache Performance by Exploiting Read-Write Disparity. In *HPCA*, 2014.
- [48] S. Khan et al. The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study. In *SIGMETRICS*, 2014.
- [49] S. Khan et al. PARBOR: An Efficient System-Level Technique to Detect Data Dependent Failures in DRAM. In *DSN*, 2016.
- [50] Y. Kim et al. Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors. In *ISCA*, 2014.
- [51] Y. Kim et al. ATLAS: A scalable and high-performance scheduling algorithm for multiple memory controllers. In *HPCA*, 2010.
- [52] Y. Kim et al. Thread Cluster Memory Scheduling: Exploiting Differences in Memory Access Behavior. In *MICRO*, 2010.
- [53] Y. Kim et al. A Case for Exploiting Subarray-Level Parallelism (SALP) in DRAM. In *ISCA*, 2012.
- [54] E. Kultursay et al. Evaluating STT-RAM as an energy-efficient main memory alternative. In *ISPASS*, 2013.
- [55] B. Lee et al. Phase-Change Technology and the Future of Main Memory. In *IEEE Micro*, 2010.
- [56] B. C. Lee et al. Architecting Phase Change Memory As a Scalable DRAM Alternative. In *ISCA*, 2009.
- [57] B. C. Lee et al. Phase Change Memory Architecture and the Quest for Scalability. In *CACM*, 2010.
- [58] C. J. Lee et al. DRAM-Aware Last-Level Cache Writeback: Reducing Write-Caused Interference in Memory Systems. In *UT Tech Report TR-HPS-2010-002*, 2010.
- [59] C. J. Lee et al. Prefetch-Aware DRAM Controllers. In *MICRO*, 2008.
- [60] C. J. Lee et al. Prefetch-Aware Memory Controllers. In *IEEE TC*, 2011.
- [61] C. J. Lee et al. Improving Memory Bank-level Parallelism in the Presence of Prefetching. In *MICRO*, 2009.
- [62] D. Lee et al. Simultaneous Multi-Layer Access: Improving 3D-Stacked Memory Bandwidth at Low Cost. In *ACM TACO*, 2016.
- [63] D. Lee et al. Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case. <http://www.ece.cmu.edu/~safari/tools/alDRAM-hpca2015-fulldata.html>.
- [64] D. Lee et al. Adaptive-latency DRAM: Optimizing DRAM timing for the common-case. In *HPCA*, 2015.
- [65] D. Lee et al. Tiered-latency DRAM: A low latency and low cost DRAM architecture. In *HPCA*, 2013.
- [66] D. Lee et al. Decoupled Direct Memory Access: Isolating CPU and IO Traffic by Leveraging a Dual-Data-Port DRAM. In *PACT*, 2015.
- [67] J. Lee et al. Simultaneously Formed Storage Node Contact and Metal Contact Cell (SSMC) for 1Gb DRAM and Beyond. In *IEDM*, 1996.
- [68] Y. Li et al. Managing Hybrid Main Memories with a Page-Utility Driven Performance Model. In *CoRR abs/1507.03303*, 2015.
- [69] J. Liu et al. An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms. In *ISCA*, 2013.
- [70] J. Liu et al. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *ISCA*, 2012.
- [71] S. Liu et al. Hardware/software techniques for DRAM thermal management. In *HPCA*, 2011.

- [72] D. Lo et al. Heracles: Improving resource efficiency at scale. In *ISCA*, 2015.
- [73] Y. Lu et al. High-Performance and Lightweight Transaction Support in Flash-Based SSDs. In *IEEE TC*, 2015.
- [74] Y. Luo et al. WARM: Improving NAND flash memory lifetime with write-hotness aware retention management. In *MSST*, 2015.
- [75] J. Meza et al. A Case for Small Row Buffers in Non-Volatile Main Memories. In *ICCD, Poster Session*, 2012.
- [76] J. Meza et al. A large-scale study of flash memory failures in the field. In *SIGMETRICS*, 2015.
- [77] Micron. RLDram 2 and 3 Specifications. <http://www.micron.com/products/dram/rlDRAM-memory>.
- [78] T. Moscibroda and O. Mutlu. Memory Performance Attacks: Denial of Memory Service in Multi-core Systems. In *USENIX Security*, 2007.
- [79] T. Moscibroda and O. Mutlu. Distributed Order Scheduling and Its Application to Multi-core Dram Controllers. In *PODC*, 2008.
- [80] S. P. Muralidhara et al. Reducing memory interference in multicore systems via application-aware memory channel partitioning. In *MICRO*, 2011.
- [81] O. Mutlu. Memory Scaling: A Systems Architecture Perspective. In *IMW*, 2013.
- [82] O. Mutlu. Memory Scaling: A Systems Architecture Perspective. In *MemCon*, 2013.
- [83] O. Mutlu et al. Address-value delta (AVD) prediction: increasing the effectiveness of runahead execution by exploiting regular memory allocation patterns. In *MICRO*, 2005.
- [84] O. Mutlu et al. Techniques for efficient processing in runahead execution engines. In *ISCA*, 2005.
- [85] O. Mutlu et al. Efficient Runahead Execution: Power-efficient Memory Latency Tolerance. In *IEEE Micro*, 2006.
- [86] O. Mutlu and T. Moscibroda. Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors. In *MICRO*, 2007.
- [87] O. Mutlu and T. Moscibroda. Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems. In *ISCA*, 2008.
- [88] O. Mutlu et al. Runahead execution: an alternative to very large instruction windows for out-of-order processors. In *HPCA*, 2003.
- [89] O. Mutlu et al. Runahead execution: An effective alternative to large instruction windows. In *IEEE Micro*, 2003.
- [90] O. Mutlu and L. Subramanian. Research Problems and Opportunities in Memory Systems. In *SUPERFRI*, 2015.
- [91] K. Nesbit et al. AC/DC: an adaptive data cache prefetcher. In *PACT*, 2004.
- [92] K. J. Nesbit et al. Fair Queuing Memory Systems. In *MICRO*, 2006.
- [93] R. H. Patterson et al. Informed Prefetching and Caching. In *SOSP*, 1995.
- [94] G. Pekhimenko et al. Toggle-Aware Bandwidth Compression for GPUs. In *HPCA*, 2016.
- [95] G. Pekhimenko et al. Exploiting Compressed Block Size as an Indicator of Future Reuse. In *HPCA*, 2015.
- [96] G. Pekhimenko et al. Linearly Compressed Pages: A Low-complexity, Low-latency Main Memory Compression Framework. In *MICRO*, 2013.
- [97] G. Pekhimenko et al. Base-Delta-Immediate Compression: A Practical Data Compression Mechanism for On-Chip Caches. In *PACT*, 2012.
- [98] M. Qureshi et al. Enhancing lifetime and security of PCM-based main memory with start-gap wear leveling. In *MICRO*, 2009.
- [99] M. Qureshi et al. AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems. In *DSN*, 2015.
- [100] M. K. Qureshi et al. Adaptive Insertion Policies for High Performance Caching. In *ISCA*, 2007.
- [101] M. K. Qureshi et al. A Case for MLP-Aware Cache Replacement. In *ISCA*, 2006.
- [102] M. K. Qureshi et al. Scalable High Performance Main Memory System Using Phase-change Memory Technology. In *ISCA*, 2009.
- [103] S. Raoux et al. Phase-change random access memory: A scalable technology. In *IBM Journal of Research and Development*, 2008.
- [104] S. Rixner et al. Memory Access Scheduling. In *ISCA*, 2000.
- [105] Y. Sato et al. Fast Cycle RAM (FCRAM); a 20-ns random row access, pipe-lined operating DRAM. In *Symposium on VLSI Circuits*, 1998.
- [106] V. Seshadri et al. The Dirty-Block Index. In *ISCA*, 2014.
- [107] V. Seshadri et al. Fast Bulk Bitwise AND and OR in DRAM. In *IEEE CAL*, 2015.
- [108] V. Seshadri et al. RowClone: Fast and Energy-efficient in-DRAM Bulk Data Copy and Initialization. In *MICRO*, 2013.
- [109] V. Seshadri et al. Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-unit Strided Accesses. In *MICRO*, 2015.
- [110] V. Seshadri et al. The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing. In *PACT*, 2012.
- [111] A. Shafiee et al. MemZip: Exploring Unconventional Benefits from Memory Compression. In *HPCA*, 2014.
- [112] Y. H. Son et al. Reducing Memory Access Latency with Asymmetric DRAM Bank Organizations. In *ISCA*, 2013.
- [113] S. Srinath et al. Feedback Directed Prefetching: Improving the Performance and Bandwidth-Efficiency of Hardware Prefetchers. In *HPCA*, 2007.
- [114] L. Subramanian et al. The Blacklisting Memory Scheduler: Achieving high performance and fairness at low cost. In *ICCD*, 2014.
- [115] L. Subramanian et al. The Blacklisting Memory Scheduler: Balancing Performance, Fairness and Complexity. In *TPDS*, 2016.
- [116] L. Subramanian et al. The Application Slowdown Model: Quantifying and Controlling the Impact of Inter-Application Interference at Shared Caches and Main Memory. In *MICRO*, 2015.
- [117] L. Subramanian et al. MISE: Providing Performance Predictability and Improving Fairness in Shared Main Memory Systems. In *HPCA*, 2013.
- [118] H. Usui et al. DASH: Deadline-Aware High-Performance Memory Scheduler for Heterogeneous Systems with Hardware Accelerators. In *ACM TACO*, 2016.
- [119] R. Venkatesan et al. Retention-Aware Placement in DRAM (RAPID): Software Methods for Quasi-Non-Volatile DRAM. In *HPCA*, 2006.
- [120] F. Ware and C. Hampel. Improving Power and Data Efficiency with Threaded Memory Modules. In *ICCD*, 2006.
- [121] P. R. Wilson et al. The Case for Compressed Caching in Virtual Memory Systems. In *ATEC*, 1999.
- [122] H.-S. Wong et al. Metal Oxide RRAM. In *Proceedings of the IEEE*, 2012.
- [123] H.-S. Wong et al. Phase Change Memory. In *Proceedings of the IEEE*, 2010.
- [124] D. Yaney et al. A meta-stable leakage phenomenon in DRAM charge storage - Variable hold time. In *IEDM*, 1987.
- [125] H. Yoon et al. Efficient Data Mapping and Buffering Techniques for Multilevel Cell Phase-Change Memories. In *ACM TACO*, 2014.
- [126] D. Zhang et al. TOP-PIM: Throughput-oriented Programmable Processing in Memory. In *HPCA*, 2014.
- [127] T. Zhang et al. Half-DRAM: A high-bandwidth and low-power DRAM architecture from the rethinking of fine-grained activation. In *ISCA*, 2014.
- [128] Y. Zhang et al. Frequent value locality and value-centric data cache design. In *ASPLOS*, 2000.
- [129] Z. Zhang et al. Cached DRAM for ILP Processor Memory Access Latency Reduction. In *IEEE Micro*, 2001.
- [130] J. Zhao et al. FIRM: Fair and High-Performance Memory Control for Persistent Memory Systems. In *MICRO*, 2014.
- [131] H. Zheng et al. Mini-rank: Adaptive DRAM architecture for improving memory power efficiency. In *MICRO*, 2008.